813-RD-013-001

EOSDIS Core System Project

Planning and Scheduling Prototype Results Report for the ECS Project

October 1995

Hughes Information Technology Corporation
Upper Marlboro, Maryland

Planning and Scheduling Prototype Results Report for the ECS Project

October 1995

Prepared Under Contract NAS5-60000

APPROVED BY

A. Anders /s/	10/31/95
Tony Anders, FOS Segment Manager	Date
EOSDIS Core System Project	

Hughes Information Technology Corporation
Upper Marlboro, Maryland

813-RD-013-001

This page intentionally left blank.

Preface

This document contains the prototype results report for phase 4 of the Planning and Scheduling prototype, which was performed between February 1995 and August 1995. Planning and Scheduling is part of the Flight Operations Segment within the EOSDIS Core System project.

This document is an informal document that is approved at the ECS Office Manager level and does not require Government approval. After this prototype has been completed, a final report will be documented in the Prototyping and Studies Final Report, DID 331/DV3.

For additional technical information pertaining to the Planning and Scheduling prototype, contact Bill Moore, Planning and Scheduling Section Manager at 301-925-0378 or via electronic mail BILLM@EOS.HITC.COM.

This page intentionally left blank.

Abstract

The purpose of the Planning and Scheduling (P&S) Prototype Results Report is to present the design and existing feature of the P & S phase 3 prototype, in addition to analyzing the science and flight operations community feedback to establish future priorities. Throughout the phase 3 prototyping effort, operational requirements and user feedback obtained from the phase 1, phase 2 and informal phase 3 demonstrations contributed to the prototype design. The design was subsequently developed into an evaluation package for presentation to the science and flight operations community at the ECS PRR, February 23, 1995. During this evaluation phase, feedback from the science and flight operations community was solicited through evaluation of informal demonstrations, meetings and audience issues and comments raised at the ECS PRR. The feedback was reviewed and incorporated into recommendations for phase 3 of the P&S prototype effort and helped establish priorities for the overall development effort.

Keywords: prototyping, evolvability, IST, ASTER, P&S, CERES, MOPITT, PIs, TLs, TDRSS, FOS, EOC

This page intentionally left blank.

Change Information Page

List of Effective Pages	
Page Number	**Issue**
Title	Original
iii through x	Original
1-1 through 1-4	Original
2-1 through 2-34	Original
3-1 through 3-10	Original
4-1 through 4-2	Original
A-1 through A-16	Original
AB-1 and AB-2	Original

Document History			
Document Number	**Status/Issue**	**Publication Date**	**CCR Number**
813-RD-013-001	Original	October 1995	

This page intentionally left blank.

Contents

4. Summary

Figures

Tables

Appendix A. P & S Class Hierarchies

Abbreviations and Acronyms

1. Introduction

1.1 Identification

The purpose of the Planning and Scheduling (P & S) Prototype Results Report is to present the design and existing features of the P & S phase 4 prototype, in addition to analyzing the science and flight operations community feedback to establish future priorities. Throughout the phase 4 prototyping effort, operational requirements and user feedback obtained from the phase 3 Prototype Results Review (PRR) and informal phase 4 demonstrations contributed to the prototype design, which was developed into an evaluation package for presentation to the science and flight operations community at the FOS PRR, August 24, 1995. During this evaluation phase, feedback from the science and flight operations community was solicited through evaluation of informal demonstrations, meetings, and audience issues and comments raised at the FOS PRR. The feedback was reviewed and incorporated into recommendations for phase 4 of the P & S prototype effort, and helped establish priorities for the overall development effort.

1.2 Prototype Overview

The P & S prototype effort is an evolutionary process of software design and development for the operational P & S system. Each phase of the prototype can be considered an incremental development towards the final release packages. Below is the schedule of activities related to phases 1-4 of the P & S prototype development effort.

Phase 1

Prototype Requirements Definition	5/93 - 6/93
Prototype Design	5/93 - 7/93
Ops Concept Development	5/93 - 8/93
Prototype Development	5/93 - 10/93
Formal Demonstration	11/93
Prototype Results Report	1/94

Phase 2

Prototype Requirements Definition	11/93 - 1/94
Prototype Design	11/93 - 1/94
Ops Concept Development	11/93 - 3/94
Prototype Development	12/93 - 5/94
Formal Demonstration	6/94
Prototype Results Report	8/94

Phase 3

Prototype Requirements Definition	6/94 - 7/94
Prototype Design	6/94 - 8/94
Ops Concept Development	6/94 - 9/94
Prototype Development	6/94 - 12/94
Formal Demonstration	1/95
Prototype Results Report	2/95

Phase 4

Prototype Requirements Definition	3/95 - 4/95
Prototype Design	3/95 - 5/95
Ops Concept Development	3/95 - 6/95
Prototype Development	3/95 - 7/95
Formal Demonstration	8/95
Prototype Results Report	10/95

By providing incremental prototypes, the science and flight operations community can offer feedback that influences the design and architecture throughout all stages of project development. For example, the user feedback obtained from the Phase 3 PRR was evaluated and helped establish the areas of focus for the phase 4 design.

1.3 Objectives

One of the primary objectives of the development effort was to refine the framework of the P & S system. This framework includes an architecture for providing distributed planning and scheduling. The software design would establish an evolvable system that allows future spacecraft and operational concept changes.

Analyzing the science and flight operations community feedback has the objective of establishing the guidelines and priorities that need to be incorporated into later phases of prototype development and the design of the system. Comments and issues raised by the science and flight operations community were acquired in various ways, including informal demonstrations, evaluation surveys, meetings, and documented questions and comments from the FOS PRR.

1.4 Applicable Documents

The following documents are applicable to the material in this document.

193-317-DV1-001	ECS Prototype Plan
193-318-DV3-005	ECS Prototype Plan & Progress Report
193-707-PP1-002	ECS Prototype Results Review
193-216-SE1-001	ECS Requirements Specification

193-604-OP1-001	ECS Operations Concept Document for the ECS Project
194-207-SE1-001	ECS System Design Specification
194-813-SI4-002	Planning and Scheduling Prototype Results Report for the ECS Project (Phase 1)
194-813-SI4-002	Planning and Scheduling Prototype Results Report for the ECS Project (Phase 2)
194-813-SI4-002	Planning and Scheduling Prototype Results Report for the ECS Project (Phase 3)
	P & S Prototype Design Specification
	EOS Distributed Planning and Scheduling Prototype Lessons Learned Working Paper
	Technical White Paper on the Hughes Inter-Process Communication Library (HIPC)

This page intentionally left blank.

2. Results

2.1 Prototype Driver Analysis

Throughout all phases of the prototyping effort, operational requirements and mission needs drive the design and development of the system. For the P & S prototype, the drivers were derived from several sources and were used to shape the overall prototyping strategy. The drivers originated from the following sources:

a) **System Requirements**

   There exist several key requirements that impose a critical risk or significant cost impact, such as the distribution performance needs of the EOC and ISTs.

b) **Phase 3 Prototype Results Review Feedback**

   During the phase 3 PRR, user feedback was gathered from the science and flight operations community through evaluation surveys, meetings and documenting questions and comments. Topics focused on remaining infrastructure performance problems, key areas of confusion after the Phase 3 prototype, and problems relating to the transition from design to development.

c) **GSFC EOS Distributed P & S Prototype Lessons Learned**

   The GSFC EOS Distributed P & S Prototype Lessons Learned document represents the findings from a coordinated study between the University of Colorado, JPL and GSFC that investigated issues related to distributed P & S. Based on the lessons learned, the primary topics included the necessity for a clear understanding of activity definitions and parameters, as well as difficulties in developing a scheduling system with external P & S components (e.g. ASTER ICC).

d) **Hughes Mission Planning IR&D**

   The Hughes Mission Planning IR&D offers lessons learned from previous prototyping experience, such as human-machine interface (HMI) approach, architecture and interactive scheduling.

e) **Previous Operations Experience**

   Based on previous experience with operational systems, evolving P & S concepts have been established from operation support. For example, experience on long-term projects has shown that the operations concept is constantly evolving as the operators learn new ways to use the system. Therefore, there is a need to support an operations concept that will change throughout the mission life.

f) **Previous Mission Planning Development Experience**

   Through development of previous mission planning systems, design elements have been identified that are critical to mission performance. For example, the interface with command management will have a direct impact on the P & S system.

The result of the driver analysis and the projected response phase is summarized in Table 2-1.

Table 2-1. P & S Prototype Phases Response Approach (1 of 2)

		Response Plan			
Prototype Driver	**Origin**	**Phase I**	**Phase II**	**Phase III**	**Phase IV**
Provide the capability to support an evolving baseline of spacecraft and instruments	System requirements	Develop a flexible, extensible object class structure that will support future system upgrades and changes. Evaluate alternative structures.	Evaluate and refine model for AM-1 to demonstrate approach.	Develop initial comm system model and evaluate extensibility.	Refinement of activities that will be able to model commanding requests for various instruments in a more generic manner.
Provide a system that can support an evolving Operations Concept	Experience, system requirements, Phase 1 PRR Feedback	Design and evaluate alternatives that support ease of upgrade, ease of run-time reconfiguration, and ease of code modification.	Demonstrate and evaluate operations flexibility with AM-1 spacecraft and instrument manifest.	Demonstrate and evaluate evolvability via initial end-to-end flow through the system.	Present the end-to-end operations concept with most tools and interfaces prototyped.
Provide a mission planning system that supports a distributed P & S Operations Concept	System requirements, GSFC Distributed Test Bed Lessons Learned, Phase 1 PRR Feedback	Demonstrate global visibility of constraints and plans based on a distributed system architecture	Refine distributed resource model concepts	Further refine the distributed resource model concepts and evaluate distributed P & S alternatives	Evaluate distributed resource models including various design testing.
Provide for simplicity of operation and common look-and-feel. Minimize training and maintenance cost.	Experience, system requirements, Hughes Mission Planning IR&D lessons learned, Phase 1 PRR Feedback	Develop a set of common planning tools that can be used across the EOC and IST. Initial phase 1 prototype will have a basic timeline and resource model.	Include additional functionality into the timeline and add other toolset components to simplify scheduling analysis.	Refine spacecraft and instrument resource models. Include additional toolset components.	Add functionality to tools initially prototyped in the Phase 3 prototype to simplify scheduling including access control.

Table 2-1. P & S Prototype Phases Response Approach (2 of 2)

Prototype Driver	**Origin**	**Response**			
		Phase I	**Phase II**	**Phase III**	**Phase IV**
Provide an early P & S test bed to mitigate external interface risks (e.g. ASTER interface, command management interface)	Experience, GSFC Distributed Test Bed Lessons Learned	Develop activity generator to be used as a basis for an ASTER interface test driver.	Develop and refine interface issues and concepts. Establish ASTER interface test driver	Develop and refine FOS internal interfaces (e.g. Command Management System).	Develop and refine FOS internal and initiate FOS external interfaces (e.g. FDF, NCC)
Provide a framework for the P & S development	Experience, proposed evolutionary development methodology	Build prototype to operations code standards and documentation. Phase 1 will include a simple end-to-end thread that will provide the basis for subsequent phases to build upon.	Incremental prototype design and development baseline	Incremental prototype design and development baseline	Experiment the transition from design to development to identify early any problem areas.

Additional drivers and modifications to the planned responses may be identified upon further analysis and feedback .

2.2 Prototype Design Analysis

Based on the prototype drivers (see Section 2.1), several analyses were performed during the phase 4 prototype to address issues related to the ECS P & S design, architecture and development. The findings of these investigations during the phase 4 prototype were based upon several sources, including:

- Informal prototype demonstrations to the user community
- Conversations with the user community
- ECS Level 3 and Level 4 Requirements
- Phases 1, 2, and 3 PRR Feedback
- GSFC EOS Distributed P & S Prototype Lessons Learned
- Previous Hughes experience in developing mission planning systems

Table 2-2 breaks down the primary design decisions related to the ECS P & S system. The overall details of the design and architecture are explained in Section 2.3.

Table 2-2. P & S Prototype Design Issues

Design Issue	**Criteria**	**Decision**	**Rationale**
Distributed P & S	In order to allow simultaneous phases of schedule development over a distributed network, scheduling access should be controlled by time periods (initial scheduling, final scheduling, etc.) and resource type (CERES, MODIS, etc.).	Plan Tool and Timeline Processes	The plan tool and timeline processes will control read/write access to the mission planning database, allowing a distributed instrument community to work concurrently on different phases of schedule development.
Mitigate Risks	Provide the distribution of scheduling data between geographically distributed locations fulfilling the performance needs defined within the Level 4 Requirements.	Data Distributor Process	The data distributor process will be run as a companion to the resource model process in order to handle the distribution of plan updates throughout the system. The previous design architectures failed under testing of a larger number of distributed planning sites.
	Reduce the development effort required going from the design of the system into full-scale development. Eliminate the risk involved in getting the main infrastructure developed first.	Early determination of code development dependencies, use prototype code whenever possible.	By establishing the coding dependencies early, it will eliminate coders waiting for another piece of the system to be completed. In addition, for the majority of coding, the skeleton classes produced by the StP/OMT CASE tool will suffice, but a large amount of time will be saved by salvaging those portions of the prototype code that match the design.
Design Issue	Criteria	Decision	Rationale
Scheduling Methodolog y	Allow users to define dynamically the constraints violations that might occur during the scheduling of predefined activities including those constraints that might arise between activities and modes, and activities and events.	Use existing prototype code.	Several options were considered in handling the determination of constraint violations during scheduling. Two COTS products, ILOG and BPARR, offered the most functionality and compatibility, but were deemed unusable due to exceedingly high cost, considerably training, and a difficult integration. The prototype code developed for handling constraints would take minimal modification for use in the final system and would cost less.

2.3 Prototype Architecture

2.3.1 P & S Architecture Design

One of the primary objectives of the phase 4 prototype was to continue refining the framework for the P & S system. Based upon the analysis of the requirements and user needs (see Sections 2.1-2.2), the already existing architecture was refined. The P & S architecture consists of independent, distributed C++ processes that communicate by passing object oriented messages between each other. Figure 2-1 shows the overall P & S architecture for the phase 4 prototype. The architecture embodies concepts and design elements from the EOS Distributed Prototype, other Hughes developed mission management systems and the phases 1, 2 and 3 PRR feedback. Each of the current P & S processes are described below:

2.3.1.1 Resource Model

The resource model is the portion of the P & S system that represents those aspects of the real world necessary for mission planning. The process models the behavior of mission resources, such as the instruments, spacecraft and subsystems. Because it simulates the behavior of these mission elements, the resource model performs a level of constraint checking based upon any physical limits. In addition, the resource model maintains the state of the instruments, subsystems and other elements. Other processes, such as the instrument activity scheduler, can modify and update the states by interfacing with the resource model.

2.3.1.2 Activity Definer

The activity definer is a tool which provides mission planners with the ability to define the types of activities they will schedule for a particular instrument or subsystem. Activity definitions contain the mode of the resource during and after the activity as well as any necessary commands and parameter defaults. The user will later be able to schedule one of the activities over a given time range using the instrument activity scheduler (see Section 2.3.1.7).

2.3.1.3 BAP Definer

In order to simplify routine scheduling tasks, users define Baseline Activity Profiles (or BAP's) using the BAP definer. A BAP is a group of activities which can be scheduled as a single activity using the instrument activity scheduler (see Section 2.3.1.7). The BAP definer builds the BAP's that the user will need by adding activities to and removing activities from a specified BAP definition. The user sets start and stop times for the activities in that BAP using times relative to the BAP's scheduled times or relative to orbit events.

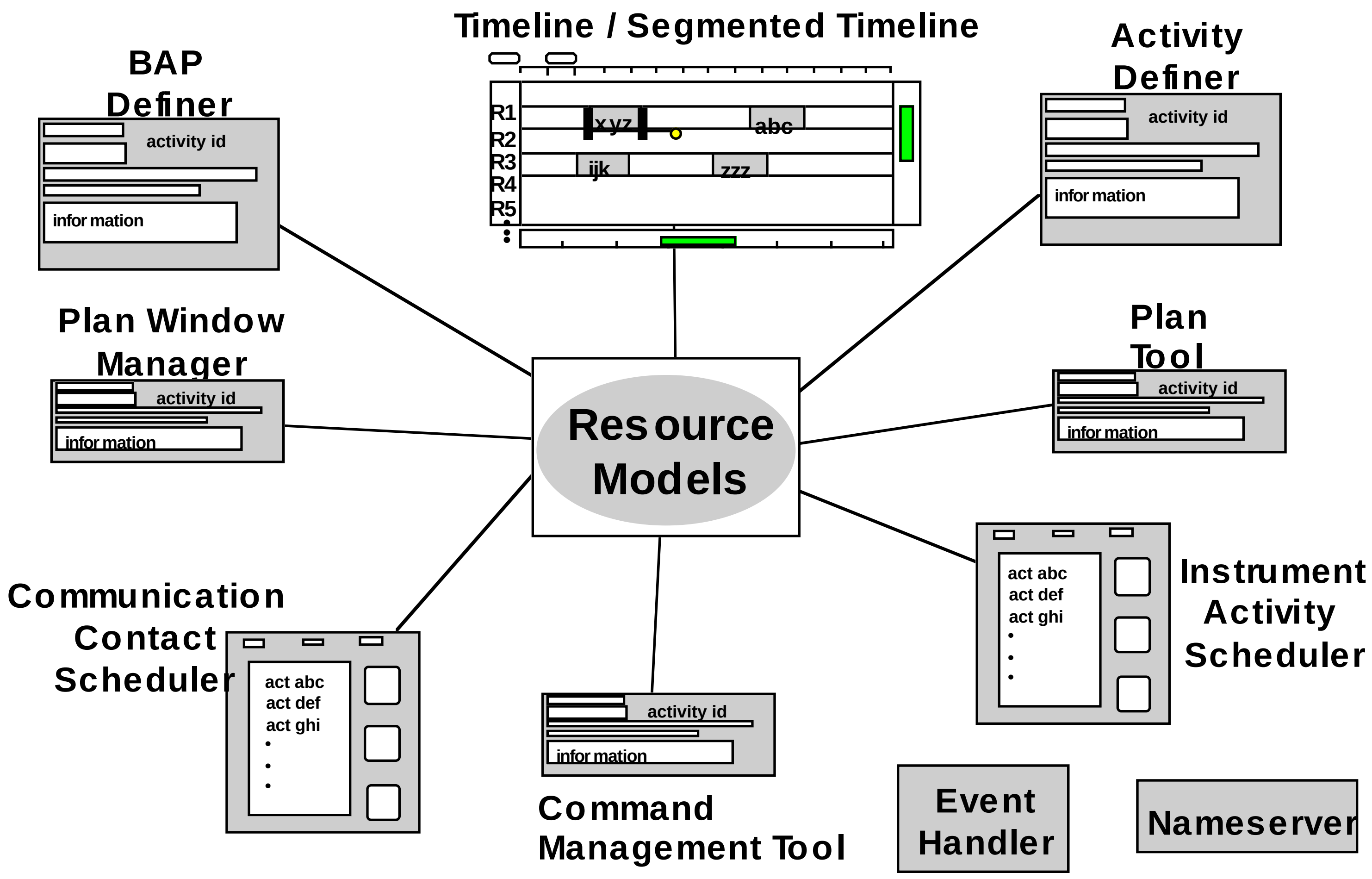

Figure 2-1. P & S Architecture

2.3.1.4 Timeline

The timeline process presents a graphical representation of spacecraft and instrument activities as a function of time. Activities are the building blocks from which mission plans are constructed, representing the state of an instrument, subsystem or other schedulable entity. The timeline displays information regarding activity start/stop times, resource consumption and constraints. By receiving and displaying schedule updates from the resource model, a timeline represents the current state of the overall mission plan. The timeline also has the capability to show user accesses, graphical representations of reserving time periods over resources. The resource reservations allow users to schedule on the resources without conflicting with each other.

2.3.1.5 Segmented Timeline

The segmented timeline process behaves similarly to the regular timeline in that it displays a graphical representation of spacecraft and activities as a function of time. However, unlike the regular timeline, the segmented timeline allows the user to view resources for consecutive days on a single display. This allows the user to view activities that may be tied to particular orbit events that may occur on a daily basis. For example, if an instrument planner wishes to view a repetitive activity that may be triggered by a daily orbit event, the segmented timeline will be able to give him the capability to view the daily repetition by specifying a certain number of days to be displayed in 24 hour sections.

2.3.1.6 PostScript Timeline

The PostScript Timeline tool is a version of the Planning & Scheduling timeline that produces PostScript output suitable for printing on black & white or color PostScript printers, or for displaying using a PostScript Viewer. The PostScript timeline uses the same configuration files as the regular timeline, so the resources, colors and text are similar. The PostScript timeline does not need the resource scroll bar, time scale scroll bar, or window borders. Printed PostScript timelines are constrained by the resolution of the printer (typically 300 dpi) as opposed to the resolution of the video screen (typically 72 dpi). Screen snapshots also include unnecessary scrollbars and window borders. The prototype version prints on a single page in landscape mode, scaling the resources & time scale to fill the page. This tool produces superior quality output that can be printed or displayed at the EOC, IST's or any other site with PostScript capabilities.

2.3.1.7 Instrument Activity Scheduler

Planning and Scheduling needs to provide the ability to schedule activities for instruments. Activities that can be scheduled are either pre-defined activities or pre-defined lists of activities called baseline activity profiles (BAPs). Once these activities or BAPs have been defined, the user needs to enter the appropriate data and press the schedule button. An instance of an activity will be created and given to the resource model to be scheduled. The scheduling algorithms schedule activities sequentially and depending on user needs, an impact or non-impact scheduling algorithm is used. The software uses different types of scheduling algorithms which can easily be changed from one type to another. Once the resource model performs the scheduling, the results can be seen on the timeline.

2.3.1.8 Communication Contact Scheduler

The Communication Contact Scheduler tool is used for establishing TDRSS and ground station contact periods. The Communication Contact Scheduler chooses desirable contact requests from a list of availability times. The availability times are the inview times minus previous rejections. The scheduler can be used for either the initial list of requests to send to the NCC or for subsequent iterations with NCC. For the PRR demonstration, inviews for TDRSS were supplied by the NCC and read in by the Communication Contact Scheduler. The data used in the demonstration was for TDRS-1, but the scheduler could handle data for any relay or ground resource with appropriate inview periods.

The algorithm the prototype scheduler uses to determine the best contacts is a score-based depth first search. The scheduler reads the score ranges from a configuration file for properties such as separation time (time between end of previous contact and start of this contact), contact duration, predicted data volume at start of contact, amount of data loss if this contact is missed and the communication path used (i.e., TDRS-E, DSN, WSGT). For the current prototype, only the separation, duration and communication path properties were implemented. The algorithm accepts several command line arguments that affect the depth of the search, the granularity of the search and a pruning parameter. The depth controls how many contacts to look ahead when scheduling a contact. No look-ahead (depth=1) works fine as long as there are no large gaps in the availability periods. Using depth=3 does a good job of placing contacts immediately before & after large gaps. The granularity specifies the difference between subsequent attempts. The demonstrations used a granularity of 60 seconds. The last parameter is a pruning parameter and it controls whether the algorithm searches paths that are worse than the best. The pruning parameter is a percentage between 0 (do an exhaustive search) and 100 (only use Look-ahead to break ties). Values around 70-90 seem to be a good mix between execution speed and finding optimal contacts.

2.3.1.9 Plan Tool

The plan tool is a user interface program that allows the user to create multiple plans. This capability will be needed if an instrument planner wishes to create a "what-if" plan to validate how changes to his instrument schedule will affect other instruments or spacecraft subsystems. In addition to plan creation, the plan tool allows the user to delete plans, copy plans (or portions of a plan) and snap to plans. The snap function is used to cause other tools that display information about a plan to automatically change to the specified plan. For example, if a user wishes to snap to a master plan and visibility process such as a timeline was currently displaying activities on a "what-if" plan, the timeline would automatically switch and immediately display the activities and spacecraft resources on the master plan.

2.3.1.10 Plan Window Manager

In order to allow simultaneous phases of schedule development over a distributed network, the plan window manager will control access to the mission planning database. The ability to make changes to the schedule will be controlled in two different ways. First, the plan window manager will manage schedule development over time periods. This will allow the planners and schedulers to simultaneously work on different time sections of the plan which is necessary to perform functions related to initial scheduling and final scheduling. Second, the plan window manager will control schedule development over specific resources, such as the CERES and MODIS instruments. Resource control will allow the distributed instrument community to simultaneously schedule their instruments over the same time period without impacting one another. Resource control will also prevent one instrument team from changing another team's schedule.

2.3.1.11 Activity Interface (Filter)

The Planning and Scheduling Subsystem interacts with nine interface elements. The context diagram in Figure 2-2 represents these elements, along with their corresponding data flows.

To handle the different types of data transactions, interface processes will exist between the P & S subsystem and the interface elements. For the purpose of prototyping, a general purpose filter was developed to allow the ingest of ASCII formatted information to the resource model. The current prototype uses an activity interface to ingest an ASCII representation of an ASTER ICC activity list. Each entry in the ASCII list represents an ASTER activity to be scheduled.

2.3.1.12 CMS Interface

The interface between planning and scheduling and command management provides constraint checking during activity scheduling and generates a detailed activity schedule, an ATC load and a ground script. When an activity is scheduled, the information is passed to command management and is expanded into commands and put into a schedule of commands. Once the commands have been expanded into the schedule, constraint checking is performed and a message is sent back to planning and scheduling notifying them if any constraints have been violated. When the user wants to generate the detailed activity schedule, the user will enter a start and stop time and then press the Generate Load button. A message with the detailed activity schedule is then sent to command management where an ATC load and ground script are generated from their schedule of commands.

2.3.1.13 IST Interface

During this phase of the prototype, a first attempt to integrate the P & S system into the Instrument Support Toolkit (IST) was made. The IST provides users with the ability to enter "rooms" where different tasks are performed. For example, the user may evaluate real-time data from a TDRSS contact in one room and schedule new contacts in the P & S room. In this prototype, both the IST and the P & S system were modified in order that the P & S system would be presented in one of the rooms of the IST.

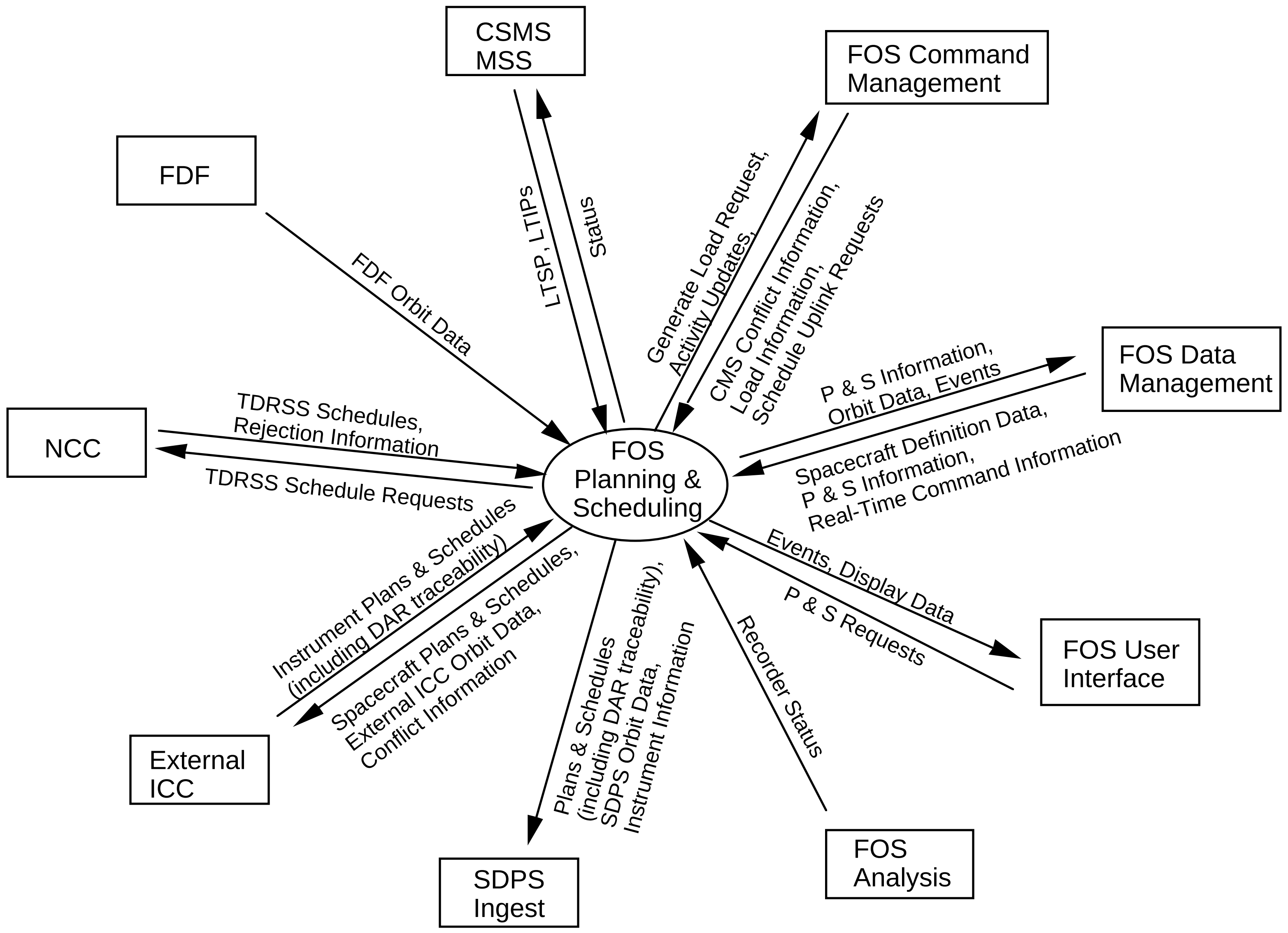
CSMS MSS
FOS Command Management
FDF
FOS Data Management
NCC
FOS Planning & Scheduling
FOS User Interface
External ICC
SDPS Ingest
FOS Analysis
FDF Orbit Data
LTSP, LTIPs
Status
Generate Load Request, Activity Updates,
CMS Conflict Information, Load Information, Schedule Uplink Requests
P & S Information, Orbit Data, Events
Spacecraft Definition Data, P & S Information, Real-Time Command Information
TDRSS Schedules, Rejection Information
TDRSS Schedule Requests
Instrument Plans & Schedules (including DAR traceability)
Spacecraft Plans & Schedules, External ICC Orbit Data, Conflict Information
Plans & Schedules (including DAR traceability), SDPS Orbit Data, Instrument Information
Recorder Status
Events, Display Data
P & S Requests

Figure 2-2. P & S Subsystem Context Diagram

2.3.1.14 Analysis Interface

The interface between planning and scheduling and analysis provides a mechanism through which analysis sends planning and scheduling information concerning the real time status of the solid state recorders. Planning and scheduling shows the status of the solid state recorders on the timeline by a line graph. The initial data that planning and scheduling uses to show this status is predicted data. To show the users the most accurate data, planning and scheduling displays as-flown data such as actual playback times and buffer contents provided by real-time analysis.

Analysis creates a connection with planning and scheduling process and sends planning and scheduling a message containing a start time associated with the start of a playback for the solid state recorder buffers and the number of EDU blocks that were played back for each buffer. Then the information can be displayed on the timeline, giving the user a graphical view of the solid state recorder status.

2.3.1.15 Name Server

Since the P & S architecture consists of separate processes that communicate by message passing, these processes must have the ability to locate one another. A name server performs this function, allowing processes to exist at remote locations for distributed planning and scheduling. When a process, such as a timeline, is started within a work group, it registers address information with the name server. A process that wishes to establish communication will ask the name server for the necessary address information. Name servers can be constructed to establish connections with other remote name servers, allowing work groups to establish communications between themselves. For the ECS P & S system, the name server has been extended to provide hooks for supporting DCE.

2.3.2 Future P & S Architecture

Although not included in the phase 4 prototype, other processes were identified based upon the design analysis (see Section 2.2) and incorporated into the P & S architecture. The following processes are necessary for EOS mission planning, and may be incorporated into later prototype developments.

2.3.2.1 Constraint Definer

After defining activities, the user will need the capability to define the resource configurations during which the execution of an activity would be hazardous. These constraints will be defined using the Constraint Definer Tool, which will allow users to the specify an activity, mode or event constrained by a different activity, mode or event. The types of constraints provided in the current design are Before, After, During and Contains. The constraints may also be offset by some interval (e.g. Activity A must be before Activity B by more than 20 seconds, where 20 seconds is the offset). The user-defined constraints would be saved to the database and could be modified at a later date if needed.

2.3.2.1 Event Scheduler

The FDF event scheduler is responsible for ingesting FDF data product files and incorporating the data within the planning and scheduling subsystem. Planning and scheduling will use FDF data for capabilities such as scheduling activities that start relative to an orbit event, providing available TDRSS scheduling windows for scheduling NCC contacts or using the predicted attitude data for determining local modes for the MISR instrument. FDF data (such as predicted ephemeris) will also be stored in the database maintained by the DMS subsystem for use by the Analysis subsystem, and will be sent to the SDPS (Science Data Processing Segment) for use in determining data acquisition times. FDF data will also be sent to the ASTER ICC for use in scheduling the ASTER instrument.

2.3.2.2 Load Scheduler

The FOS will provide the capability to schedule the uplink times for the various spacecraft and instrument loads, such as microprocessor loads and table loads. The Load Scheduler Tool will schedule an uplink window that indicates the time period the user would prefer the load to be sent. The time period may correspond to a specific communication contact (e.g. one TDRSS contact) or a longer time duration (e.g. a 24 hour time period). The FOT will use the uplink window for choosing a communication contact upon which the load will be sent. The user will be notified of the chosen communication contact through the timeline display.

2.3.2.3 Activity Recycler

The Activity Recycler Tool will allow the FOT/IOT to perform queries on the mission plan for specific activity types (e.g. MODIS Calibration Activities) that have been removed from the schedule. The Activity Recycler will display the list of activities previously removed from the schedule and, if desired, save the list for future reference. This allows the user to make changes to the mission plan and, for any reason, reschedule the removed list of activities at a later time. The Activity Recycler will also receive the list of activities that were removed during the generation of the conflict-free Detailed Activity Schedule. Once a saved list is no longer needed, the user may delete it with the Activity Recycler.

2.3.3 Inter-Process Communication

The distributed P & S architecture consists of separate processes that communicate by message passing. To accomplish inter-process communication, a set of software objects known as the Hughes Inter-Process Communication (HIPC) class libraries are utilized to provide peer-to-peer communications between concurrently executing processes. These processes may be running on the same computer or distributed across a heterogeneous computer network. If used with C++ processes, HIPC allows objects to be passed between processes. For external processes outside of P & S that are written in other languages (e.g. C, Ada or FORTRAN), interface handlers will facilitate the communication. For a more detailed description on the inter-process communication approach, refer to the technical paper on the Hughes Inter-Process Communication Library referenced in section 1.4.

2.3.4 Mission Planning Class Libraries

The P & S architecture is based upon the Hughes Mission Planning Class Libraries, represented by the software usage hierarchy diagram shown in Figure 2-3. The mission planning heritage code is a collection of C++ class libraries which provide baseline classes for display, inter-process communication and P & S. These class libraries are developed by Hughes and embody concepts and design elements from other Hughes developed mission management systems. As represented in Figure 2-3, the ECS P & S system is built as an extension to the Mission Planning Class Libraries. By taking advantage of tested code that encompasses mission management experience, development and test cycles for the ECS P & S system are shortened, thereby reducing overall life cycle costs (see Section 2.2) .

At the foundation of the software usage hierarchy are the Hughes Class Libraries (HCL). HCL is a collection of C++ class libraries that are used as general purpose programming utilities. HCL includes class libraries for handling collections, displays, and inter-process communication. The Hughes Mission Planning Class Libraries are built on top of HCL. These libraries include:

- **Timeline Class Libraries (TCL)—**framework for displaying time ordered information on a timeline graph
- **Scheduling Class Libraries (SCL)—**framework for development of system domain specific scheduling algorithms
- **Resource Class Libraries (RCL)—**framework for modeling space based, ground based and related resources
- **Planning Class Libraries (PCL)—**framework for the planning and scheduling of mission resources (built on top of TCL, SCL and RCL)

For configuration management, all of the heritage class libraries, along with the extensions built for the phase 3 P & S prototype, use the Source Code Control System (SCCS). Because ECS is currently establishing a project-wide configuration management system under a new product, ClearCase, SCCS will be replaced during the Phase 4 prototype.

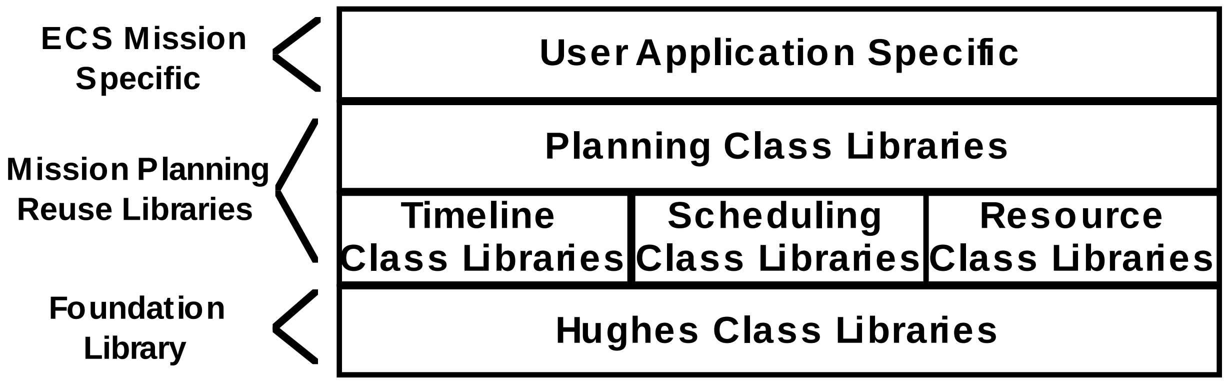

Figure 2-3. Mission Planning Class Libraries

For a more detailed description of HCL and the Hughes Mission Planning Class Libraries, refer to the P & S Prototype Design Specification.

2.3.5 Evolvability

The P & S development effort is an evolving process, where each prototype phase is considered an incremental development towards the operational system. Throughout the development process, the P & S system must be able to accept changes in the mission requirements and operational concepts. To accomplish this, the C++ programming language was utilized to take advantage of an object oriented development approach.

Object oriented programming allows the P & S problem domain to be separated into software objects (instances of classes) that encapsulate the attributes and behavior of the physical mission elements. Figure 2-4 shows the P & S class structure representing an EOS spacecraft and its components. At the bottom of the class structure are software objects that represent the physical spacecraft elements, such as the AM-1 Payload and the CERES instrument. These software objects offer a direct mapping into the problem domain, so when the science and flight operations community speaks of a spacecraft element, it corresponds to a software object that encapsulates its attributes and behavior. Encapsulation minimizes the impact due to an evolving operational concept, since changes to the requirements of a physical mission element will only affect its corresponding software object.

In addition to encapsulation, object oriented development allows inheritance of attributes and behavior. For example, the CERES class shown in Figure 2-4 inherits the attributes and behavior of its parent class ECSSimpInstr. The CERES class represents an extension to the more general parent class. As specific mission requirements become known, inheritance allows P & S extensions to be built on top of the existing knowledge contained in the class structure.

Figure 2-5 shows the object oriented representation of the AM-1 spacecraft. The AM-1 spacecraft object consists of a payload, instrument and subsystem objects. To encapsulate behavior, each instrument and subsystem object control their own allocation. The AM-1 payload object controls the interfacing between the instrument objects and the subsystem objects. For instance, when the MODIS instrument object needs subsystem resources (power, data, etc.), it asks the AM-1 payload object, which in turn asks the necessary subsystems. By encapsulating the instrument and subsystem objects in this manner, changes to instrument or subsystem operations will only affect the corresponding software object. In addition, hooks were put in place for future elements by creating objects without incorporating the specific attributes and behavior (e.g. DAS).